الفصل الخامس ٥

برمجة عائلة المتحكمات PIC

Programming PIC Microcontrollers

٥-١ مقدمة

بعد أن فهمنا التركيب الداخلى للمتحكمات البسيطة والمتوسطة الإمكانيات يبقى أمامنا أن نبرمج هذه الشرائح لعمل تطبيق معين . بالطبع فإن التطبيقات تختلف من حيث الصعوبة والسهولة من تطبيق لآخر ، ولكن جميع التطبيقات التى تستخدم المتحكمات من أى نوع ومن أى شركة ينقسم التعامل معها إلى جزأين :

- **الجزء الأول** : هو بناء الدائرة الإلكترونية Hardware بما فى ذلك المتحكم نفسه والدوائر المحيطة به التى ستقوم إما بتهيئة الإشارة الداخلة قبل قراءتها بالمتحكم ، أو الدوائر التى ستأخذ خرج المتحكم وتهيئه بحيث يتناسب مع الدائرة الخارجية التى سيتم إدارتها بالمتحكم مثل إدارة موتور ، أو تشغيل لاقط relay أو مظهر display من أى نوع . بالطبع فإن خطوة مهمة فى هذا الجزء هى اختيار المتحكم المناسب للتطبيق الذى سيتم التعامل معه .
- **الجزء الثانى** : هو كتابة البرنامج الذى سيتم تسجيله داخل ذاكرة البرمجة الخاصة بالمتحكم بحيث أنه عند تشغيل المتحكم أو توصيل القدرة للنظام يتم تنفيذ هذا البرنامج . كتابة البرنامج لأى متحكم ، مهما كان هذا البرنامج بسيطا ، من المستحيل أن ياتى صح مائة بالمائة من أول محاولة بحيث أنك بعد كتابة البرنامج على الورقة ثم تسجيله على ذاكرة البرمجة داخل المتحكم ثم توصيل القدرة ستفاجأ بأن النظام لا يعمل ولا يعطى الخرج أو التحكم المطلوب . هنا عليك إعادة النظر فى البرنامج والبحث فيه عن أى أخطاء وتصحيحها ثم إعادة تسجيل البرنامج مرة أخرى . فى العادة عملية التسجيل والتصحيح وإعادة التسجيل تأخذ عدة محاولات مما يتطلب مجهود ووقت قد يطول . لذلك فإن كل الشركات المصنعة للمتحكمات تقوم بتصميم برنامج محاكاة simulation لهذه المتحكمات حيث يتم تثبيت هذه البرامج على الحاسب العادى pc ومن ثم يمكن لمستخدم شريحة المتحكم أن ينفذ كل برامجه على برنامج المحاكاة على جهاز الحاسب وبعد التأكد من صحة البرنامج وتشغيله بالكامل ، فى هذه الحالة نقوم بتثبيته أو تسجيله على ذاكرة البرمجة فى شريحة المتحكم مرة واحدة فقط .

لذلك فإن تنفيذ أى تطبيق باستخدام أحد شرائح المتحكمات يتم بالخطوات التالية :

١- تنفيذ الدائرة Hardware

٢- تصميم البرنامج بعد عمل مخطط سير له flow chart ثم كتابته على ورقة باللغة المناسبة .
٣- تنفيذ البرنامج على برنامج المحاكاة simulator حتى يتم التأكد من صحته ووضعه فى صورته المثلى .
٤- تثبيت أو تسجيل البرنامج على ذاكرة البرمجة لشريحة المتحكم باستخدام دائرة خاصة بذلك توصل على الحاسب من خلال المخرج المتوازى parallel port أو المخرج المتوالى serial port . فى العادة تباع هذه الدائرة مع برنامج المحاكاة .

إن شركة Microchip المصنعة لمتحكمات PIC قد قامت بعمل برنامج المحاكاة الخاص بها وهو البرنامج MPLAB الذى يمكن به تنفيذ أى برنامج لأى متحكم من منتجات هذه الشركة . لحسن الحظ فإن هذا البرنامج متاح على موقع الشركة يمكن تنزيله مجانا وبدون أى تكلفة . موقع الشركة على الإنترنت هو www.microchip.com . سنقدم فى هذا الفصل شرحا سريعا لبرنامج المحاكاة MPLAB وسنستخدمه فى تنفيذ برامج الأمثلة المقدمة فى هذا الفصل .

٥-٢ برنامج المحاكاة MPLAB

هذا البرنامج بجانب كونه برنامج محاكاة يمكن به تنفيذ برامج منتجات PIC كلها ، فإنه أيضا يمكن من خلاله تسجيل البرنامج على شريحة المتحكم . هذا البرنامج يعمل على كل إصدارات windows المتاحة . شكل (٥-١) يبين شاشة الدخول فى برنامج MPLAB حيث نلاحظ من هذا الشكل أن شاشة الدخول تحتوى المكونات القياسية لشاشات الدخول فى جميع البرمجيات التى تعمل تحت نظام النوافذ windows . هناك شريط للقوائم (١) menu bar ، ثم شريط للأيقونات (٢) icon bar ، ثم مساحة بيضاء يمكن فيها تحرير البرامج .

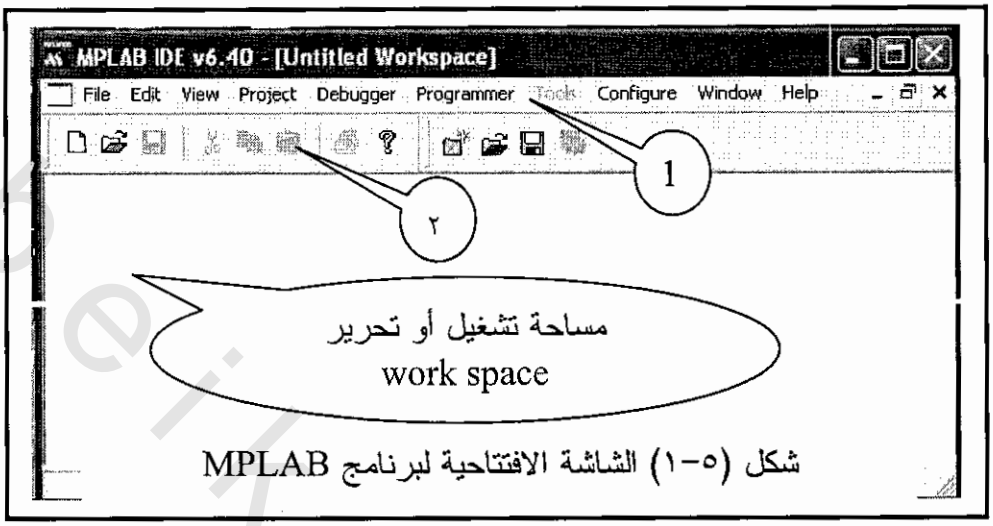
أحسن وسيلة سريعة نفهم بها كيفية التعامل مع هذا البرنامج هى من خلال شرح تفصيلى لمثال بسيط .

مثال ٥-١

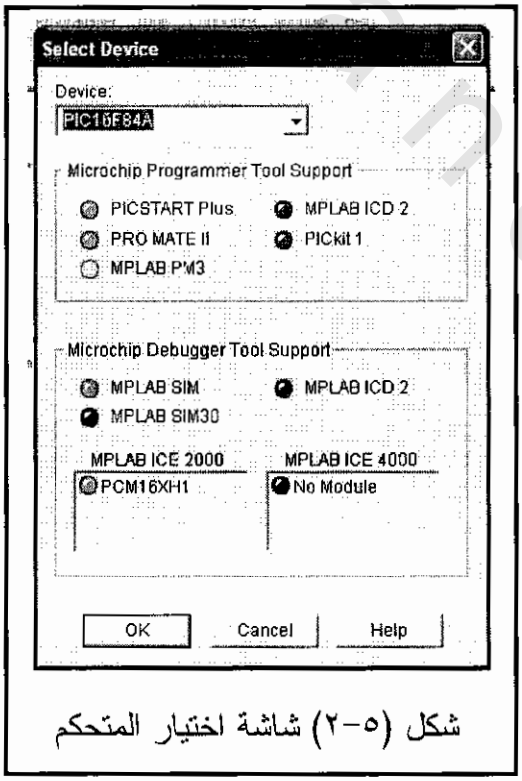
فى هذا المثال سنضع أى رقم ، ٣٠ مثلا ، فى أى واحد من المسجلات العامة ، المسجل 0x10 مثلا ، ثم نظل ننقص محتويات هذا المسجل بمقدار واحد إلى أن يصبح صفر فيقف البرنامج . سنتبع الخطوات التالية فى كتابة وتنفيذ البرنامج :

١. نبدأ أول خطوة باختيار نوع أو رقم المتحكم الذى سنتعامل معه فى هذا البرنامج من ضمن القائمة الكبيرة من المتحكمات التى تنتجها هذه الشركة . لعمل ذلك أنقر على قائمة configure فى شريط القوائم ثم انقر على الخيار Select Device حيث ستظهر لك شاشة جانبية بها قائمة كاملة بكل منتجات PIC المتاحة والتى

يمكنك التجوال خلالها حتى تجد المتحكم الذى تريده وعندها أنقر عليه ليتم اختياره ثم انقر OK . فى هذا المثال اخترنا المتحكم 16f84A كما فى شكل (٥-٢) .



شكل (٥-١) الشاشة الافتتاحية لبرنامج MPLAB



شكل (٥-٢) شاشة اختيار المتحكم

بعد اختيار المتحكم الذى سنتعامل معه فى هذا البرنامج ستظهر لك شاشة كالموضحة فى شكل (٥-٢) حيث تلاحظ أن هذه الشاشة تبين البرنامج الذى يمكن استخدامه لحرق أو تسجيل برنامجنا على شريحة المتحكم . Microchip Programmer tool Support حيث البرنامج الذى أمامه علامة خضراء هو فقط الذى يمكن استخدامه فى ذلك مثل برنامج PICSTART Plus كما فى الشكل . هناك أيضا برمجيات التصحيح والتتبع مثل برمجية MPLAB SIM وهو البرنامج الذى سنستخدمه لتصحيح وتتبع تنفيذ البرنامج Debugger .

٢. فى الخطوة الثانية سنفتح المحرر الملحق على البرنامج MPLAB ونبدأ فى كتابة البرنامج الخاص بنا . يتم

الدخول فى المحرر من قائمة File ثم النقر على New حيث سيفتح البرنامج مساحة التحرير التى سنبدأ بكتابة البرنامج فيها .

٣. الآن سنكتب البرنامج الخاص بالمثال السابق والذى يضع الرقم ٣٠ مثلا فى المسجل رقم 10 ثم نبدأ فى إنقاصه إلى أن يصل للصفر فيقف البرنامج . بعد كتابة هذه الأوامر يصبح البرنامج كالتالى من داخل المحرر :

```
;*******************************************************
;
;       First trial program .
;
;*******************************************************

    list    p=16F84A          ; list directive to define processor
        #include <p16F84A.inc>      ; processor specific variable definitions

        __CONFIG  _CP_OFF & _WDT_ON & _PWRTE_ON & _RC_OSC

    errorlevel    -302   ;Eliminate bank warning


;***** VARIABLE DEFINITIONS

;*******************************************************************************
                cc equ 0x10
                ORG    0x000          ; processor reset vector
                goto   main           ; go to beginning of program.

main

; remaining code goes here
                org 0x06
                movlw 0x04
                movwf cc
loop            nop
                decfsz cc,1
                goto      loop
                halt

                END                   ; directive 'end of program'
```

أى سطر من أسطر هذا البرنامج يبدأ بفاصلة منقوطة معناه أن هذا السطر يمثل تعليقا ولا يعتبر أمرا من أوامر المتحكم . فى العادة نبدأ البرنامج بكتابة بعض الأسطر نكتب فيها اسم البرنامج والغرض منه ، وكذلك اسم مصمم البرنامج والشركة التابع لها إن وجد . يأتى بعد ذلك مجموعة الأسطر المظللة وفيها السطر الأول يحدد نوع المتحكم ، والثانى يضم الملف الخاص بهذا المتحكم ، والثالث خاص بضبط أداء المتحكم من حيث حماية البرنامج وتنشيط مؤقت الحراسة ومؤقت القدرة

ونوع المذبذب والرابع يمنع ظهور التحذيرات warnings التى لا تعيق تنفيذ البرنامج. هذه الأسطر يتم نقلها كما هى من نموذج خاص ببرمجة كل متحكم ولقد تم نقل هذه الأسطر من البرنامج :

MPLAB/MCHIP_Tools/TEMPLATE/CODE/ f84atemp.asm

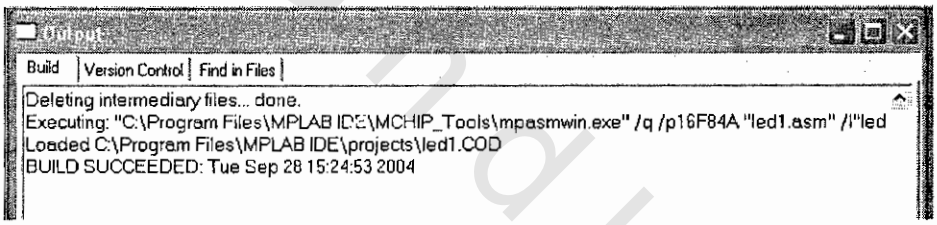
بعد ذلك يمكن كتابة برنامج التطبيق كما سبق .

بعد كتابة البرنامج نسجله فى ملف بأى اسم (proj1 مثلا) ولكن بامتداد asm. وفى أى مجلد على الحاسب . بذلك يكون قد تم تحرير البرنامج وتسجيله حيث يمكننا الآن عرضه على الأسمبلر وبرنامج التوصيل linker وذلك للحصول على نسخة قابلة للتنفيذ من البرنامج .

٤. من قائمة Project أنقر على الخيار Quickbuild proj1.asm حيث فى الحال يتم فتح نافذة بها رسائل خطأ توضح الأخطاء الموجودة فى البرنامج إن وجد . أما إذا لم يكن هناك خطأ فى البرنامج فتظهر رسالة فى هذه النافذة توضح ذلك كما فى شكل (٥-٣) . أهم ما فى هذه الشاشة هى العبارة الموجودة فى آخر سطر :

BUILD SUCCEEDED: Tue Sep 28 15:24:53 2004

والتى توضح أنه قد تم بناء البرنامج القابل للتنفيذ بنجاح وتعطى تاريخ لذلك كما فى الشكل .



شكل (٥-٣)

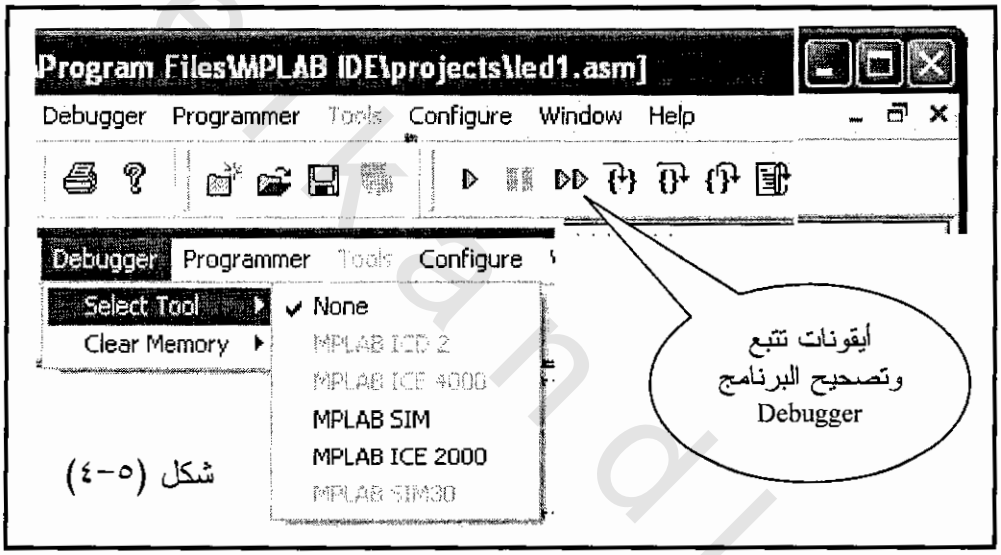
بالحصول على الصورة القابلة للتنفيذ من البرنامج يمكن الآن تتبع تنفيذه لنرى إن كان سيعطى النتائج المطلوبة منه أم لا ، أى إن كان سينفذ صح أم لا . لاحظ أن الحصول على الصورة القابلة للتنفيذ ليس معناه أن البرنامج سينفذ صح ويعطى النتائج المطلوبة منه ، ولكن ذلك يعنى فقط أن البرنامج لا يحتوى أخطاء من وجهة نظر لغة التجميع فقط . لذلك فإننا ننصح بتتبع تنفيذ البرنامج لنرى مدى صحته .

٥. من قائمة Debugger أنقر على الخيار select tools ثم منها أنقر على الخيار MPLAB SIM حيث فى الحال ستضاف أمامك قائمة من الأيقونات فى شريط الأيقونات ، هذه القائمة هى قائمة تتبع وتنفيذ البرنامج كما فى شكل (٥-٤) . هذه القائمة تحتوى العديد من الأيقونات التى تنفذ البرنامج كله مرة واحدة Run وأيقونة لوقف عملية التنفيذ Halt ، وأخرى للتنفيذ خطوة بخطوة Step into ،

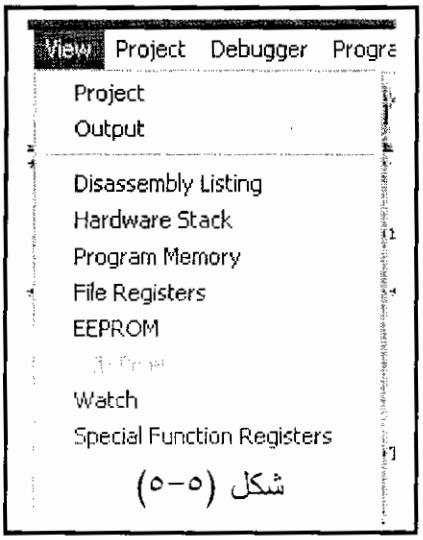
وأخرى للتنفيذ خطوة بخطوة دون الدخول فى البرامج الفرعية ولكنها تعتبر خطوة واحدة Step over وأخرى لعمل Reset للبرنامج . بهذه الأيقونات يمكن تتبع البرنامج وتنفيذه بالكيفية المطلوبة والتى بها يمكن استخراج أى أخطاء تنفيذية فى البرنامج . يمكن أيضا وضع نقاط توقف أو نقاط نظام Break points عند أى أمر فى البرنامج بحيث يتم تنفيذ بالبرنامج بالطريقة العادية إلى أن يصل لهذه النقطة فيقف .

من قائمة View يمكن عرض محتويات المسجلات الخاصة والعامة فى أثناء تنفيذ البرنامج خطوة بخطوة لنرى هل يسير البرنامج فى الاتجاه الصحيح أم لا . شكل (٥-٥) يبين محتويات هذه القائمة .

الآن حاول تطبيق كل خطوات تتبع وتصحيح البرامج على البرنامج السابق .



شكل (٥-٤)



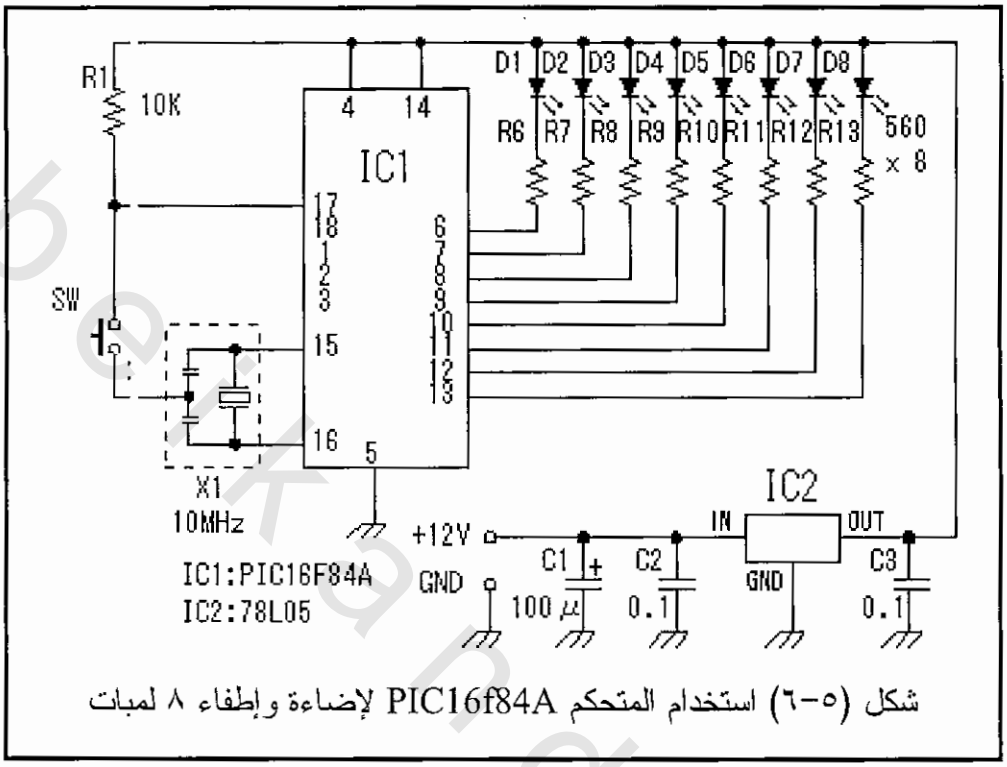
شكل (٥-٥)

فى مثال ٢ سنقدم برنامج أكثر تعقيدا فحاول تتبعه وتصحيح كل ما سبق من خطوات التنفيذ عليه .

مثال ٥-٢

سنقوم هنا بتوصيل ٨ لمبات بيان على البوابة B بعد إعدادها لتكون بوابة إخراج ثم نجعل هذه اللمبات تضىء وتطفىء كلها كل نصف ثانية طالما أن المفتاح الموصل على الطرف الأول من البوابة A موصل على الأرضى أى مغلق . معنى ذلك أن الطرف الأول من

البوابة A سيتم إعداده كخط دخل نوصل عليه مفتاح S ، عندما يكون هذا المفتاح مفتوح ، فإن اللمبات تتوقف عن الإطفاء أو الإضاءة ، وعندما يكون مغلق فإن اللمبات تضيء وتطفيء . شكل (٥-٦) يبين الدائرة المقترحة لذلك .



شكل (٥-٦) استخدام المتحكم PIC16f84A لإضاءة وإطفاء ٨ لمبات

البرنامج فكرته بسيطة حيث سيكون عبارة عن حلقة لا نهائية تقرأ الطرف A0 فإذا كان واحد فإن ذلك يعنى أن المفتاح مفتوح (غير مضغوط) وفى هذه الحالة لا تعمل شيء سوى قراءة نفس الطرف مرة أخرى . إذا تم ضغط المفتاح فإن الطرف A0 يصبح صفرا وفى هذه الحالة يتم النداء على برنامج فرعى يغير حالة اللمبات فإذا كانت مضيئة يطفأها وإذا كانت مطفأة ينيرها عن طريق الأمر xorwf PORTB,h'ff' ثم ينادى على برنامج فرعى آخر يقوم بتحقيق زمن تأخير مقداره نصف ثانية (٥٠٠ ميللى ثانية) .

مع تتبع البرنامج ستجد أن البرنامج الفرعى الذى يحقق ٥٠٠ ميللى ثانية ينادى على برنامج فرعى آخر يحقق ١٠٠ ميللى ثانية ٥ مرات . والبرنامج الذى يحقق ١٠٠ ميللى ثانية ينادى على برنامج فرعى آخر يحقق واحد ميللى ثانية ١٠٠ مرة . البرنامج الفرعى الذى يحقق ١ ميللى ثانية يحتوى حلقتين يتم تنفيذ الداخلية فيهما ٢٤٩ مرة بحيث أن مجموع دورات الأمر فى هذا البرنامج الفرعى هو ٢٥٠١ دورة أمر . بفرض أن المتحكم يعمل بنبضات تزامن مقدارها ١٠ ميجاهرتز كما فى شكل

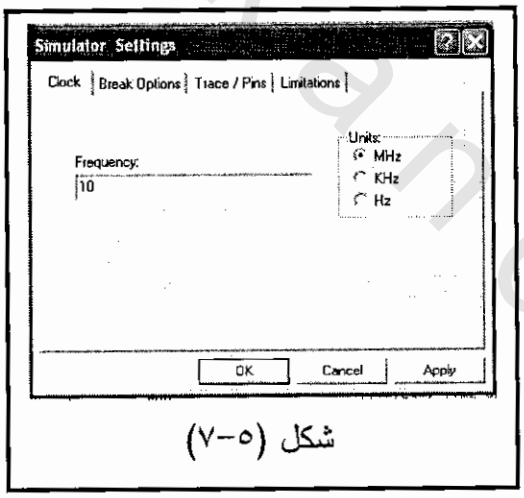
(٥-٦) فإن زمن مجموع الأوامر في هذا البرنامج الفرعي سيكون (2501x0.4usec=1msec) . قائمة أوامر البرنامج ستكون كما يلي :

```
;*****************************************************
;
;       The LED flash program .
;
;*****************************************************
    list    p=16F84A        ; list directive to define processor
        #include <p16F84A.inc>      ; processor specific variable definitions

        __CONFIG  _CP_OFF & _WDT_ON & _PWRTE_ON & _RC_OSC

    errorlevel    -302   ;Eliminate bank warning
;***************** Label Definition ********************
ra0             equ   00        ;PA0 bit
cnt500u         equ   0c        ;500usec counter Address
cnt1m           equ   0d        ;1msec counter Address.
cnt500m         equ   0f        ;500usec counter Address
cnt100m         equ   0e        ;100msec counter Address
;***************** Program Start ***********************
    org   0         ;Reset Vector
    goto  init
    org   4         ;Interrupt Vector
    goto  init
;***************** Initial Process *********************
    org   5
init
        bsf   STATUS,RP0    ;Change to Bank1
        movlw  h'ff'        ;Set input mode data
        movwf  TRISA        ;Set PORTA to Input mode
        clrf   TRISB        ;Set PORTB to Output mode
        bcf   STATUS,RP0    ;Change to Bank0
        movlw  h'ff'        ;Set LED off data
        movwf  PORTB        ;Output data.
keyscan
                btfss  PORTA,ra0     ;RA0 ON(Low lebel) ?
                call      flash
                goto keyscan
flash
                movlw  h'ff'
                xorwf   PORTB,f
                call      t500m
                return
        ;************* 500msec Timer Subroutine ***************
t500m           movlw  d'5'         ;Set loop counter
                movwf  cnt500m      ;Save loop counter
tm3lp           call   t100m        ;100msec subroutine
                decfsz  cnt500m,f     ;cnt500m - 1 = 0 ?
                goto   tm3lp        ;No. Continue
                return              ;Yes. Count end
```

```
;************* 100msec Timer Subroutine ***************
t100m           movlw  d'100'       ;Set loop counter
                movwf  cnt100m      ;Save loop counter
tm2lp           call   t1m          ;1msec subroutine
                decfsz cnt100m,f    ;cnt100m - 1 = 0 ?
                goto   tm2lp        ;No. Continue
                return              ;Yes. Count end
;************* 1msec Timer Subroutine ******************
t1m             movlw  d'2'         ;(1)    Set loop cnt1
                movwf  cnt1m        ;(1)    Save loop cnt1
tm1lp1          movlw  d'249'       ;(1)*2  Set loop cnt2
                movwf  cnt500u      ;(1)*2  Save loop cnt2
tm1lp2          nop                 ;(1)*249*2 Time adjust
                nop                 ;(1)*249*2 Time adjust
                decfsz cnt500u,f    ;(1)*249*2 cnt500u-1=0 ?
                goto   tm1lp2       ;(2)*248*2 No, continue
                decfsz cnt1m,f      ;(1)*2  cnt1m-1=0 ?
                goto   tm1lp1       ;(2)    No. Continue
                return              ;(2)    Yes. Cnt end
                                    ;Total 2501*0.4usec=1msec
                end
```



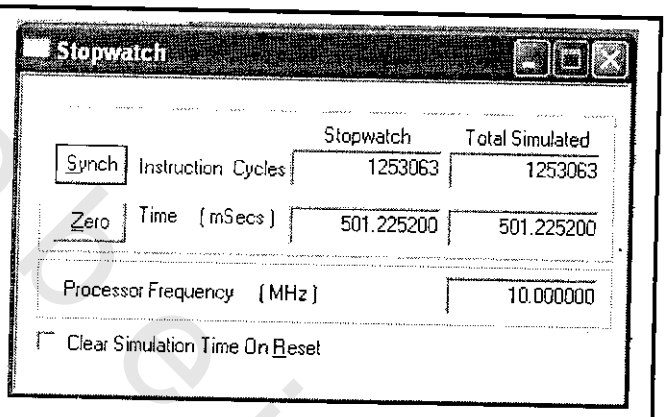
شكل (٥-٧)

حاول تنفيذ هذا البرنامج كما هو على برنامج المحاكاة MPLAB وتتبع خطواته عن طريق تنفيذه خطوة بخطوة ويمكن عن طريق ساعة الإيقاف الموجودة فى الديبجر معرفة زمن تنفيذ هذا البرنامج بالضبط . من قائمة Debugger أنقر على الخيار Settings فى آخر القائمة حيث ستفتح لك نافذة كالموضحة فى شكل (٥-٧) يمكنك منها ضبط التردد الذى سيعمل عنده المتحكم .

بعد ضبط التردد كما سبق ، من قائمة Debugger أنقر على الخيار Stopwatch حيث ستظهر لك نافذة كالموضحة فى شكل (٥-٨) . ضع نقطة توقف Break point عند مكان مناسب فى البرنامج بحيث عندما يصل التنفيذ لهذه النقطة تقف عملية التنفيذ وسترى فى شاشة ساعة الإيقاف الزمن الذى أخذه المعالج لكى يصل لهذه النقطة فى البرنامج . فى شكل (٥-٨) نلاحظ أن الزمن الذى أخذه البرنامج ليضيء أو يطفيء اللمبات فقط هو 501.2252msec أى نصف ثانية تقريبا .

يمكن وضع نقطة التوقف عند أى مكان فى البرنامج عن طريق النقر مرتين فى المسطرة الموجودة فى أقصى يسار شاشة تحرير البرنامج كما فى شكل (٥-٩)

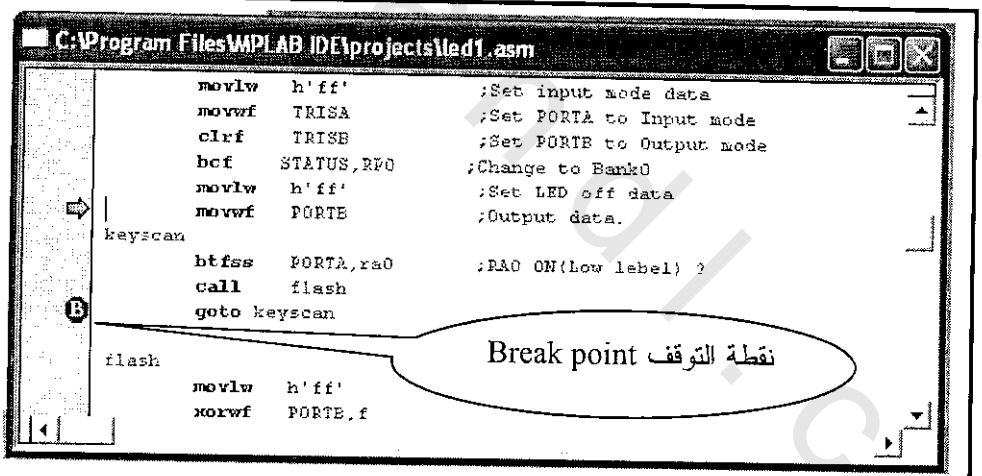
حيث تم وضع نقطة توقف عند الأمر GOTO Keyscan حيث بتنفيذ البرنامج حتى هذا الأمر والوقوف عنده يكون قد تم النداء على البرنامج الفرعى flash مرة واحدة والذى بدوره ينادى على كل برامج التأخير الفرعية .



شكل (٥-٨)

مثال ٥-٣

سنعيد نفس المثال السابق تماما ولكن هذه المرة سنحصل على زمن التأخير باستخدام المؤقت TMR0 بحيث عندما يصل المؤقت إلى القيمة FF فإنه يقاطع المتحكم ويذهب إلى العنوان 004 لينفذ برنامج خدمة القاطعة . البرنامج أصبح كالتالى :



شكل (٥-٩)

```
;*****************************************************
;
;       The LED flash program .
;
;*****************************************************

    list    p=16F84A        ; list directive to define processor
```

```
        #include <p16F84A.inc>      ; processor specific variable definitions

        __CONFIG  _CP_OFF & _WDT_ON & _PWRTE_ON & _RC_OSC

    errorlevel    -302   ;Eliminate bank warning

;*************** Label Definition ******************
ra0   equ   00          ;PA0 bit
cntr equ   0c

;*************** Program Start ***********************
    org   0           ;Reset Vector
    goto  init
    org   4           ;Interrupt Vector
                decfsz   cntr,f
                goto     xx
                movlw    h'ff'
                xorwf    PORTB,f
                movlw    d'36'
                movwf    cntr
xx              bcf      INTCON,T0IF
                clrf             TMR0
                retfie

;*************** Initial Process *********************

init
                 bsf   STATUS,RP0    ;Change to Bank1
                movlw  h'ff'         ;Set input mode data
                movwf  TRISA         ;Set PORTA to Input mode
                clrf   TRISB         ;Set PORTB to Output mode
                movlw  h'd6'
                movwf  OPTION_REG
                 bcf   STATUS,RP0    ;Change to Bank0
                movlw  h'ff'         ;Set LED off data
                movwf  PORTB         ;Output data.
                movlw  d'36'
                movwf  cntr

                clrf   TMR0
                movlw  h'a0'
                movwf  INTCON
keyscan
                btfsc  PORTA,ra0     ;RA0 ON(Low lebel) ?
                goto   xx1
                goto keyscan
xx1             halt
                end
```

برنامج خدمة المقاطعة

البرنامج الأساسى يبدأ من العلامة init حيث عندها تم تحديد البوابة A كبوابة دخـــل والبوابة B خرج ، وتم إرسال شفرة إلى المسجل Option_reg لجعل العداد القاســم

Prescaler يتبع المؤقت TMR0 وتم ضبطه بحيث يقسم على 128 وذلك بتسجيل الرقم 'h'd6 فى هذا المسجل . بعد ذلك تم تصفير المؤقت وتسجيل الرقم 'a0'h فى المسجل INTCON لتنشيط المقاطعة من المؤقت . بعد ذلك دخل البرنامج فى حلقة لانهائية يقرأ فيها الطرف RA0 ليختبره إذا كان صفر أم لا بحيث إذا كان واحد يقف البرنامج . قبل دخول البرنامج فى هذه الحلقة يتم تنشيط المؤقت ليبدأ فى العد وعندما يصل إلى FF وينزل للصفر فإنه يقاطع المتحكم وعلى الفور يذهب للعنوان 004 حيث ينفذ برنامج خدمة المقاطعة هناك . برنامج خدمة المقاطعة ينقص العداد cntr بمقدار واحد وطالما أن هذا العداد لم يصل إلى الصفر فإنه يصفر المؤقت من جديد وينشطة ليبدأ عملية العد من جديد . الهدف من العداد cntr هو تكرار عملية العد بحيث نحصل على زمن التأخير المطلوب . عندما يصل العداد cntr إلى الصفر فى برنامج خدمة المقاطعة فإنه يتم عكس محتويات البوابة B ثم تنشيط المؤقت من جديد وتحميل العداد cntr بالقيمة 36 من جديد . القيمة 36 مع القاسم 128 ونبضات التزامن ١٠ ميجاهرتز نحصل على زمن تأخير حوالى ٥٠٠ ميللى ثانية . حاول تنفيذ البرنامج السابق وتتبعة وحساب قيمة زمن التأخير الناتج .